

# Application Load Balancer Project

---

## Introduction

This project demonstrates how to deploy an **Application Load Balancer (ALB)** in AWS by launching six EC2 instances and creating three target groups. The instances represent **Home**, **Laptop**, and **Mobile** servers. The ALB is configured to distribute traffic across these targets. Finally, the DNS name of the load balancer is tested to verify traffic distribution.

---

## Overview

### 1. EC2 Instances

- Launch 6 EC2 instances:
  - Two instances for **Home**.
  - Two instances for **Laptop**.
  - Two instances for **Mobile**.
- Each instance runs Apache(httpd) web server
- Custom content is added to `index.html` using **User Data scripts**

### 2. Target Groups

- Create 3 target groups: one for two instance
- Register each EC2 instance to its respective target group

### 3. Application Load Balancer

- Create an ALB and attach the 3 target groups
  - Configure listener on **HTTP (port 80)** to forward requests
- 

## Steps to Implement

### Step 1: Launch EC2 Instances

- **AMI:** Amazon Linux
- **Instance Type:** t3.micro (Free Tier)
- Allow the inbound rules:
  - SSH (Port 22)
  - HTTP (Port 80)
- Add the following User Data scripts during launch:

#### Home Instance

```
#!/bin/bash
yum update -y
```

```
yum install -y httpd
systemctl start httpd
systemctl enable httpd
echo "<h1>This is home page $(hostname -f)</h1>" > /var/www/html/index.html
```

The screenshot shows the AWS Management Console with the URL <https://us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#LaunchInstances>. The user is in the 'Launch an instance' step of the EC2 wizard. On the left, there's a code editor window containing the provided shell script. On the right, the configuration pane shows:

- Number of instances:** 1
- Software Image (AMI):** Amazon Linux 2023 AMI 2023.8.2... [read more](#) ami-08982f1c5bf93d976
- Virtual server type (instance type):** t3.micro
- Firewall (security group):** New security group
- Buttons:** Cancel, Launch instance, Preview code

## Laptop Instance

```
#!/bin/bash
yum update -y
yum install -y httpd
systemctl start httpd
systemctl enable httpd
mkdir /var/www/html/laptop
echo "<h1>This is laptop page $(hostname -f)</h1>" >
/var/www/html/laptop/index.html
```

User data (base64 encoded):

```
#!/bin/bash
yum update -y
yum install httpd -y
systemctl start httpd
systemctl enable httpd
sudo mkdir /var/www/html/laptop
echo "<h1>This is laptop page $(hostname -f)</h1>" >
/var/www/html/laptop/index.html
```

User data has already been base64 encoded

**Summary**

Number of instances: 2

When launching more than 1 instance, consider EC2 Auto Scaling

Software Image (AMI): Amazon Linux 2023 AMI 2023.8.2... [read more](#)  
ami-08982f1c5bf93d976

Virtual server type (instance type): t3.micro

Firewall (security group)

[Cancel](#) [Launch instance](#) [Preview code](#)

## Mobile Instance

```
#!/bin/bash
yum update -y
yum install -y httpd
systemctl start httpd
systemctl enable httpd
mkdir /var/www/html/mobile
echo "<h1>This is mobile page $(hostname -f)</h1>" >
/var/www/html/mobile/index.html
```

User data - optional

Select

User data - optional | Info

Upload a file with your user data or enter it in the field.

[Choose file](#)

User data (base64 encoded):

```
#!/bin/bash
yum update -y
yum install httpd -y
systemctl start httpd
systemctl enable httpd
sudo mkdir /var/www/html/mobile
echo "<h1>This is mobile page $(hostname -f)</h1>" >
/var/www/html/mobile/index.html
```

**Summary**

Number of instances: 2

When launching more than 1 instance, consider EC2 Auto Scaling

Software Image (AMI): Amazon Linux 2023 AMI 2023.8.2... [read more](#)  
ami-08982f1c5bf93d976

Virtual server type (instance type): t3.micro

Firewall (security group)

[Cancel](#) [Launch instance](#) [Preview code](#)

## Step 2: Create Target Groups

### 1. Navigate to Target Groups

- Go to **AWS Console** → **EC2** → **Target Groups**
- Click **Create target group**

The screenshot shows the AWS Management Console interface for the EC2 service. The left sidebar is collapsed, and the main content area is titled "Target groups". A search bar at the top says "Filter target groups". Below it is a table header with columns: Name, ARN, Port, Protocol, and Target type. A message in the center states "No target groups" and "You don't have any target groups in us-east-1". At the bottom, there is a blue "Create target group" button. The footer includes copyright information and links for Privacy, Terms, and Cookie preferences.

### 2. Create Target Group for 1.Home Instance

- **Target type:** Instance
- **Protocol:** HTTP
- **Port:** 80
- **Name:** home-TG

The screenshot shows the "Create target group" wizard step 1. It has two sections: "Lambda function" (disabled) and "Application Load Balancer" (selected). The "Application Load Balancer" section contains a bulleted list: "Offers the flexibility for a Network Load Balancer to accept and route TCP requests within a specific VPC." and "Facilitates using static IP addresses and PrivateLink with an Application Load Balancer." Below this, there is a "Target group name" input field containing "Home-TG", a note about character restrictions, a "Protocol" dropdown set to "HTTP", a "Port" input field with "80", and an "IP address type" note. The footer includes copyright information and links for CloudShell, Feedback, Privacy, Terms, and Cookie preferences.

- **VPC:** Select the same VPC as your EC2 instance
- Click **Next**

### 3. Register Targets

- Select the **Home EC2 instance**
- Click **Include as pending below**

The screenshot shows the AWS Lambda console with the URL <https://us-east-1.console.aws.amazon.com/lambda/home?region=us-east-1#CreateTargetGroup>. The 'Ports for selected instances' field contains '80'. A blue button labeled 'Include as pending below' is visible.

- Click **Create target group**

The screenshot shows the AWS Lambda console with the URL <https://us-east-1.console.aws.amazon.com/lambda/home?region=us-east-1#CreateTargetGroup>. The 'Targets (2)' section lists two instances. A blue button labeled 'Create target group' is visible.

### 4. Create Target Group for Laptop Instance

- Repeat the same steps as above
- **Name:** **laptop-TG**
- Register the **Laptop EC2 instance**

### 5. Create Target Group for Mobile Instance

- Repeat the same steps again

- **Name:** mobile-TG
- Register the **Mobile EC2 instance**

The screenshot shows the AWS EC2 Target groups page. On the left sidebar, under the Instances section, there is a list of options including Instances, Instance Types, Launch Templates, Spot Requests, Savings Plans, Reserved Instances, Dedicated Hosts, and Capacity Reservations. The main content area displays a table titled "Target groups (3) Info". The table has columns for Name, ARN, Port, Protocol, and Target type. Three target groups are listed: "mobile-TG" (ARN: arn:aws:elasticloadbalancing:us-east-1:123456789012:targetgroup/mobile-TG), "Laptop-TG" (ARN: arn:aws:elasticloadbalancing:us-east-1:123456789012:targetgroup/Laptop-TG), and "Home-TG" (ARN: arn:aws:elasticloadbalancing:us-east-1:123456789012:targetgroup/Home-TG). All three target groups have a port of 80, protocol set to HTTP, and target type set to Instance. Below the table, a message says "0 target groups selected" and "Select a target group above."

## Step 3: Create Application Load Balancer

- Go to EC2 → Load Balancers → Create ALB
- Choose Application Load Balancer
- Configure:
  - Listener: HTTP (port 80)
  - Forwarding rules: Attach all 3 target groups

The screenshot shows the "Create Application Load Balancer" wizard. Under the "Routing action" section, the "Forward to target groups" option is selected. In the "Forward to target group" section, a target group named "Home-TG" is selected. This target group is defined as "Instance, IPv4" with "Target stickiness: Off". The "Weight" is set to 1 and the "Percent" is set to 100%. Other target groups listed are "Laptop-TG" and "mobile-TG", both also defined as "Instance, IPv4" with "Target stickiness: Off". A note at the bottom states: "Use stickiness the client must support cookies. If you want to bind a user's session to a specific target, turn on the Turn on target group stickiness checkbox." At the bottom of the page, there are sections for "Listener tags - optional" and "CloudShell Feedback".

- Create Application Load Balancer

The screenshot shows the AWS Management Console interface for creating a new Application Load Balancer (ALB). A green success message at the top states: "Successfully created load balancer: ALB. It might take a few minutes for your load balancer to fully set up and route traffic. Targets will also take a few minutes to complete the registration process and pass initial health checks." Below this, the "ALB" details page is displayed. Key information includes:

- Load balancer type:** Application
- Status:** Provisioning
- VPC:** vpc-0cef9cd1a366f6e90
- Hosted zone:** Z35SXDOTRQ7X7K
- Availability Zones:**
  - subnet-021eb659844bbbed2a us-east-1d (use1-az4)
  - subnet-0f50f865c18426e3b us-east-1c (use1-az2)
  - subnet-0db1b7e2e2e27857b25 us-
- Load balancer IP address type:** IPv4
- Date created:** September 17, 2025, 17:12 (UTC+05:30)

## Step 4: Test the Load Balancer

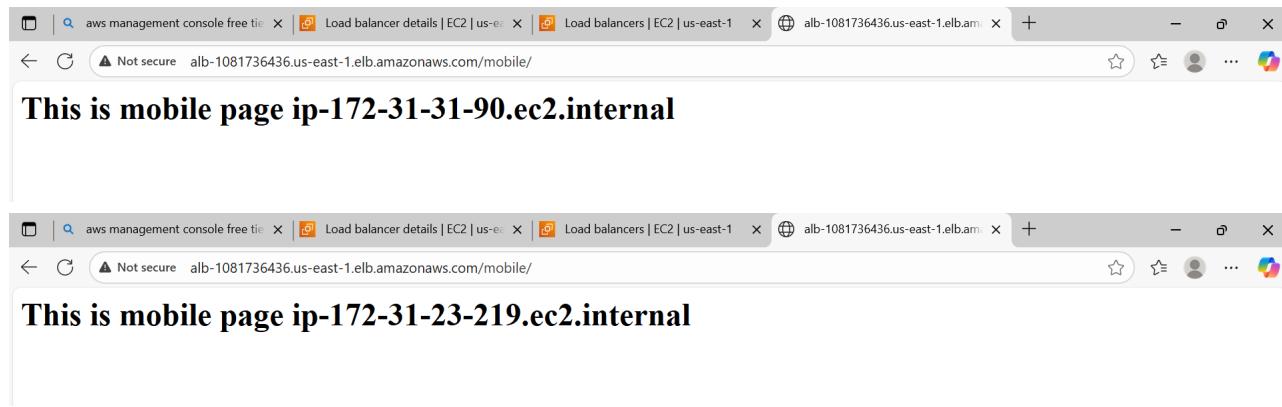
- Copy the DNS name of the ALB

A callout box highlights the copied DNS name: "ALB-1081736436.us-east-1.elb.amazonaws.com (A Record)".

- Open it in a browser and refresh multiple times
- You should see alternating responses:

The browser tabs show the ALB URL: "alb-1081736436.us-east-1.elb.amazonaws.com". The content of the browser windows alternates between:

- "This is Home page ip-172-31-22-210.ec2.internal"
- "This is laptop pageip-172-31-26-217.ec2.internal"
- "This is laptop pageip-172-31-16-131.ec2.internal"



## Summary

deployed an Application Load Balancer (ALB) to efficiently distribute incoming traffic across six EC2 instances. Each instance was configured to serve a unique webpage using Apache, with setup automated through User Data scripts. To manage routing, she created and registered three distinct target groups, ensuring organized traffic flow. The ALB listener was configured to forward requests to these target groups based on defined rules. After deployment, she verified successful load balancing by accessing the ALB's DNS name, confirming that traffic was being evenly distributed and each instance responded as expected.